Universal Modular Framework for Sensor Networks

Eli De Poorter, Benoît Latré, Ingrid Moerman and Piet Demeester

Abstract—**Wireless sensor networks are becoming increasingly popular for wireless monitoring and automation of buildings and industrial processes. Currently, many different protocols for sensor networks are proposed. However, there is very little research regarding a sensor network architecture in which to integrate them. To that end, we propose a universal, modular framework (UMF) for sensor networks. The proposed framework eases protocol integration, and has two key advantages over previous layered solutions. First, due to its modular composition, the proposed framework is highly optimized to support heterogeneous networks. Second, more advanced network functionality such as QoS is more easily supported. In this paper, we discuss several design constraints for building a modular network architecture for sensor networks and we explore several possibilities to take advantage of a modular architecture when designing network protocols.**

Index Terms—**Sensor network, cross-layer, modular architecture, sensor network architecture.**

I. Introduction

RESEARCH in wireless sensor networks has known a tremendous boost in the last few years [1]. Wireless sensor networks (WSNs) are becoming more and more widespread. Whereas sensor networks were originally used mainly for monitoring purposes, new applications such as process and asset monitoring, disaster intervention and wireless building automation are rapidly emerging.

These new and advanced applications impose new challenges and requirements on the design of WSNs. Special devices such as actors [2] are required, which can interact with the environment. Additionally, each application has its own set of specific QoS requirements, such as maximal delay, desired reliability and so on. Also, the sensors can become mobile, thus making communication more and more complex.

Considering these facts, it is clear that future sensor networks will know a wider diversity regarding the capabilities of the sensor nodes. Whereas the first sensor networks consisted of a large number of homogeneous nodes, additional computing power or functionality will be required in some. Sensor networks are thus evolving into heterogeneous networks.

Even though new applications for sensor networks are being set up frequently, there is currently no protocol framework which is widely applicable. For each layer, an application developer has to determine which protocols are most suited for the purposes of the intended application, and has to go through the complicated process of combining them into an optimized protocol stack. This comes at a great development cost, impeding the growth of sensor networks and hindering the cooperation between different sensor networks. Accordingly, there is currently a large need for a universally applicable framework for sensor networks in which existing protocols can easily be integrated.

In this paper, we propose a new universal framework for wireless sensor networks. This framework is a generic one, that can be used by different types of applications. It is designed to allow for easy integration of current protocols and takes into account the heterogeneity of the sensor nodes. Furthermore, energy-efficiency is optimally supported through cross-layer optimization. Finally, in order to support a wide range of applications, advanced functionality such as QoS and mobility can be supported.

The remainder of this paper is organized as follows. An overview of related work in the area of sensor network architectures and cross layer communication in wireless networks is given in section II. Section III gives an insight of how we see WSNs evolving towards heterogeneous networks, and how we tend to cope with this evolution using node classifications. In section IV we discuss the advantages of a modular architecture. Some challenges when designing a modular architecture are given in section V. The universal modular framework itself is discussed in section VI. In section VII we go into more detail on how to handle dependencies between modules. Finally, section VIII gives some future directions and section IX concludes the paper.

II. Related work

A myriad of protocols for sensor networks has been proposed in the last few years, taking on issues such as medium access, routing and data manipulation. Most of the early developed protocols are based on a strict layered structure [1], [3]. This way, a specific level needs no information about the inner workings of lower or higher levels. Functionalities at different layers can be altered without any impact on the other layers. This strict separation has proven to be a good solution for wired networks, allowing the protocol designers to focus only on a small subset of network functionality. However, it is not suitable for wireless networks [4]–[6]. On the other hand, the use of a cross-layer approach has several advantages: optimization is possible at several layers at once, a global optimization can be achieved and conflicts in optimizations at different layers can be avoided.

Many cross layer architectures have been proposed for ad hoc networks [4], [7]–[9] and in lesser extend for sensor networks [11]–[13]. An overview is given in figure 1. Fig. 1(a)

Manuscript received February 23, 2007. This research is partly funded by the Fund for Scientific Research - Flanders (F.W.O.-V. Belgium), by The Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen) through a PhD.grant for E. De Poorter and B. Latré and by the IBBT-WBA and IBBT-IM3 projects.

All authors are with the Department of Information Technology (INTEC - IBCN), Ghent University – IBBT vzw – IMEC vzw, Gaston Crommenlaan 8, bus 201, B-9050 Ghent, Belgium. Website: http://www.intec.ugent.be/. Phone: 0032-093314982. Fax: 0032-093314899. E-mail corresponding author: eli.depoorter@intec.ugent.be.

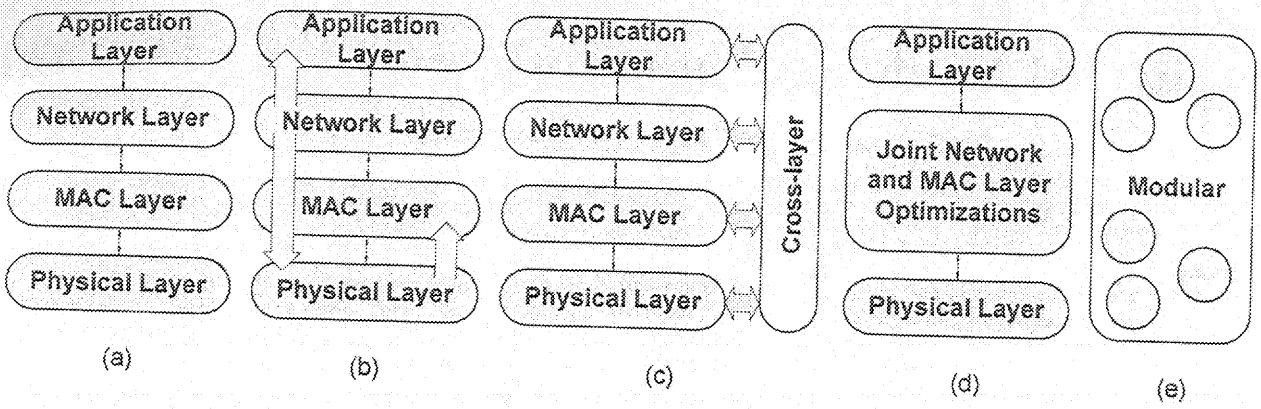


Figure 1. Overview of different cross-layer approaches. (a) Traditional layered structure (b) Passing of parameters across several layers (c) Holistic approach maintaining the different layers (d) Optimalizations spanning several layers (e) Modular approach.

shows the traditional layered structure. In the approach of Fig. 1(b), new interfaces between the existing layers are defined. An overview of protocols adopting this method can be found in [10]. A major drawback of this approach is the introduction of many dependencies between the different layers. A more holistic view is used in Fig. 1(c). A shared database is used, accessible by all the layers. Retrieval of parameters incurs additional overhead, but the resulting architecture is more clear. Another class of cross layer design is the complete merge of two or more layers of the protocol stack, e.g. the MAC-layer and the routing layer as seen in Fig. 1(d). The last approach, shown in Fig. 1(e), completely discards the layered structure. The required functionalities are implemented in different modules or heaps [14] which interact and can be changed easily.

The cross layer approach forms a first step towards a more uniform architecture for sensor networks. In order to boost the development of sensor networks, all the protocols defined for sensor networks need to be glued together . This vision is also stated in [15] where Culler et. al claim that "*the primary factor currently limiting progress in sensornets is not any specific technical challenge but is instead the lack of an overall sensor network architecture*". Therefore, they propose an overall sensor network architecture (SNA) [15] based on the sensor protocol (SP) [16], [17], which forms a *narrow waist* that sits between the network layer and link layer. SP is responsible for handling data transmission, data reception and neighbor management. However, SNA doesn't take into account the heterogeneity of the sensor nodes and has limited support for advanced network functionalities. The framework we propose will handle these problems.

III. Future sensor networks

A. Vision of future networks

As stated in section I, sensor networks are currently often considered as very large networks consisting of homogeneous nodes. In the future, these networks will evolve into heterogeneous networks where the nodes do not share the same properties. Generally speaking, we envisage 2 kinds of networks: custom made and general purpose ones.

The *custom network* is specifically designed to support one or more functions. Thus, custom networks can be highly optimized in terms of energy consumption or desired QoS. A drawback is that these networks will be expensive and only serve a single purpose. As such, little value will be given to compatibility with other protocols. This type of network will be used for applications with stringent requirements.

General purpose networks on the other hand, are more adaptable to changing network conditions and will support a wide range of applications that are unaware of underlying network conditions. These networks will have to support multiple tasks in one network. Through the use of both cheap sensor nodes and more expensive nodes which can fulfill advanced functions, considerable cost reductions can be made when deploying a sensor network. Also, in a home environment, nodes with new functionalities can be added after deployment. Since these nodes may come from different manufacturers, the network will exist of nodes with diverging capabilities. As such, great care should be taken to allow interoperability with existing protocols.

B. Node classification

As said in the section above, we envisage a network with diverse types of nodes. To prevent an unmanageable wild-growth of divergent nodes, we assume these different nodes can be categorized into several classes of nodes with similar characteristics. We assume that most sensor networks will not contain more than three classes of strongly differing nodes. However, if the need arises, the number of classes can be tailor made to the application needs following a similar logic as described in this article.

Based on their capabilities, we define 3 types of nodes: **lightweight**, **advanced** and **computing** nodes. The *lightweight nodes* have very limited resources and only support basic functionalities, hence only the basic protocols are implemented. For example, the routing protocol can be as simple as forwarding sensed data to a more advanced node. The *advanced nodes* have more sophisticated functionalities. They implement the advanced functionality which is needed for a scalable and energy-efficient WSN, such as advanced routing, clustering and data-aggregation protocols. As such,

advanced nodes typically fulfill the functions of relay nodes. The *computing nodes* are the most powerful nodes and have a much larger battery capacity (e.g. connected to the power grid) and computing power. They offer additional services, such as QoS, mobility support and network monitoring, which are not required for data gathering and relaying, but are necessary for more demanding applications. For example: a monitoring protocol can present valuable information for optimizing the routing protocol or for supporting mobility. Computing nodes thus fulfill regulating and supporting functions in addition to routing functions.

IV. Modular approach

The evolution of sensor networks as described above clearly shows the need for an optimized, generic, application independent solution in which sensor protocols can be integrated. A long-term solution should be optimized to support heterogeneity and be adaptable to future advancements such as the creation of new physical carriers. Therefore, we have defined an architecture where **functionality is divided in modules that interact with each other**. This modular approach has several advantages.

- Duplication of functionality can be avoided. Classic examples of duplicate functionalities are error correction and retransmission which are currently implemented in several layers of the protocol stack;
- Depending on the capabilities of the node, more modules (and thus network functionality) can be added. This way, heterogeneous networks can be supported;
- By allowing inter-modular parameter exchanges, cross-layer optimizations are possible. This results in much more energy-efficient protocols;
- Through the replacement of modules, it is easy to adapt to changing network conditions and future developments.

The proposed modular framework is intended to be implemented on top of an existing operating system such as TinyOS [18] or SOS [19]. We envisage a compiler environment with several implementations of each module. When faced with an application scenario, application developers can choose which modules are required for a working solution (for example, whether or not mobility or QoS support is needed). The framework should also benefit protocol developers: the interaction of the new protocol with existing network solutions can easily be investigated by choosing different implementations of the interacting modules. Thus, the freedom when choosing the appropriate modules allows for detailed fine-tuning of the framework for specific applications and allows for advanced testing of different networking aspects. Thus, various applications and networks with diverging requirements can be designed on the same foundations.

V. Challenges when designing modular protocols

Developing protocols in a modular way is a new paradigm and requires adjustments on the approach of designing network protocols. Some challenges when designing functional modules are the following:

- The parameters to be exchanged by the different modules have to be determined. The exchanged parameters should have a substantial impact on the performance of the modules and the global system. The performance increase should at least compensate for the additional complexity introduced by the cross-modular interaction [20].
- There is a need to identify which parameters need to be optimized at design-time and which at runtime. When we are optimizing at design-time, optimal operation points are calculated off-line for various predefined operational conditions. These operation points are then used at runtime. This means that a look-up table is used which maps the optimal operational points for given operational settings. Generally, runtime solutions are more flexible and hence are better suited for dynamic networks, but these will require more complex nodes, leading to larger energy consumption. Depending on the complexity of the problem and the nature of the applied modules, real-time or off-line solutions may be adopted.
- The framework can be adaptive to changing network or application conditions. E.g. it could be beneficial to have more than one routing module and to activate one of them depending on the network/application environment.
- Due to the modular design, functional and regulating modules depend on the information of other modules. However, when replacing modules, it is possible that not all of the required information is available to the framework. When designing modules, one should keep in mind how the module will react when other modules do not supply some key parameters. Possible ways to define the available parameters are handled in section VII.
- When designing modules, careful consideration should be given to mutual dependencies. In particular, circular dependencies (where a steady-state can not be obtained) should be avoided. For example, when choosing the next hop node, the decisions of the routing module (shortest path) may interfere with decisions of the QoS module (reliable link) or the energy management module (hop with most remaining battery power). To prevent unceasing competition, a priority system or arbiter (taking into account the different preferences) should be developed (see also section VI-B1). The interaction between different modules should thus be thoroughly examined.

VI. Universal modular framework

As an example of these principles, we have designed a universal modular framework (UMF) for sensor networks. A schematic overview of the universal modular framework is shown in figure 2. Generally speaking, UMF can be regarded as a hybrid combination of the architectural approaches shown in figure 1 (c) and (e). We distinguish 4 major parts in the UMF. The middle part is formed by the 'Modular Heterogeneous Sensor network Architecture' or Mohesa. This part contains the modules which implement the needed protocols for the sensor network. A 'Common Application Interface' between Mohesa and the applications is provided, which facilitates the deployment of different types of sensor networks. Further, a 'Physical Interface' is provided which can

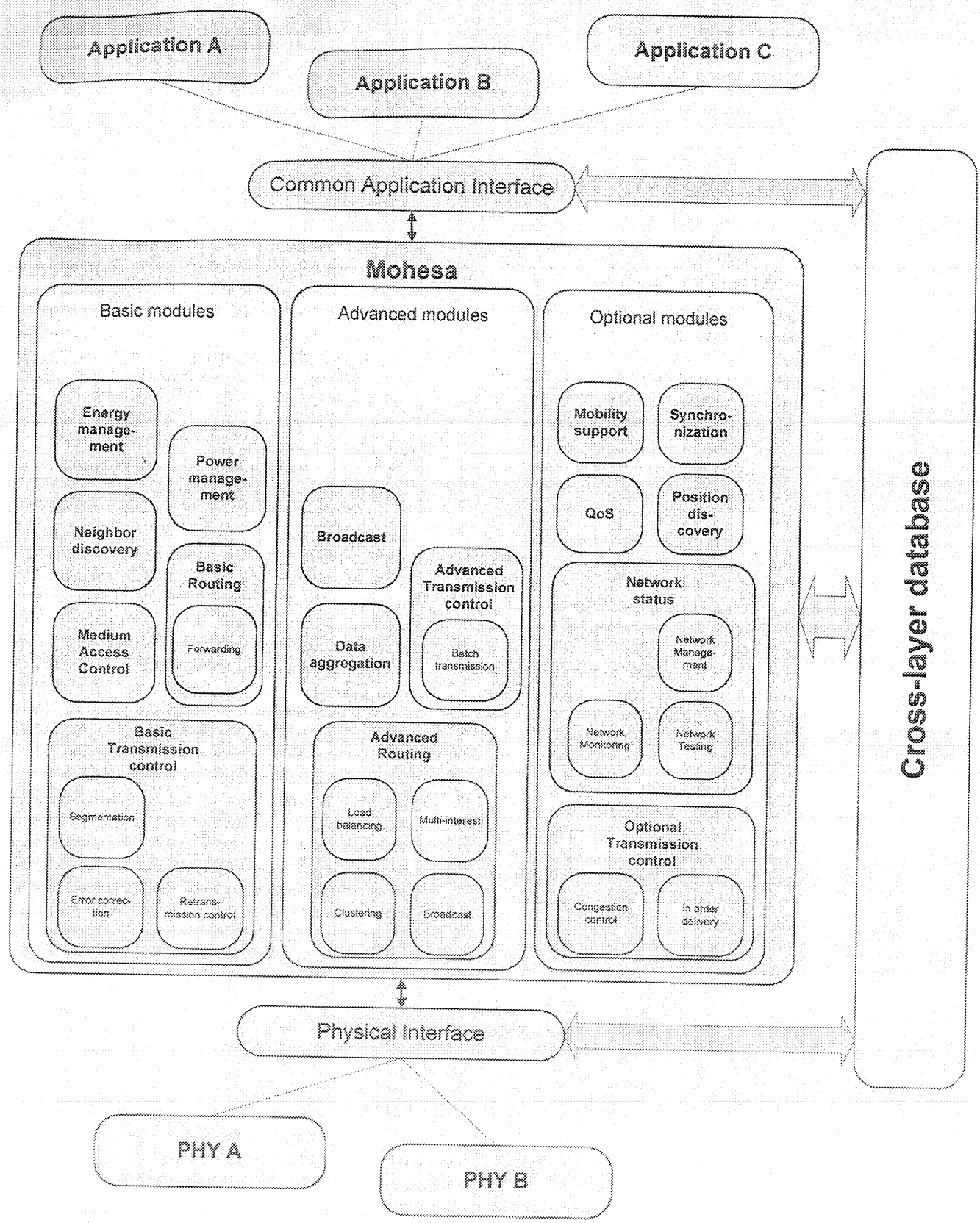


Figure 2. Proposed universal modular framework for sensor networks

be adopted to interact with various types of physical layers. And finally, a common cross-layer database is used to form a generic interface for the exchange of cross-layer parameters.

In the following sections, we give an example of a possible configuration of the sensor network, using the universal modular framework.

A. Cross-layer database

The main task of the cross-layer database is to provide a shared storage space for variables and to define a protocol that Mohesa's modules can use to exchange parameters with other modules and layers.

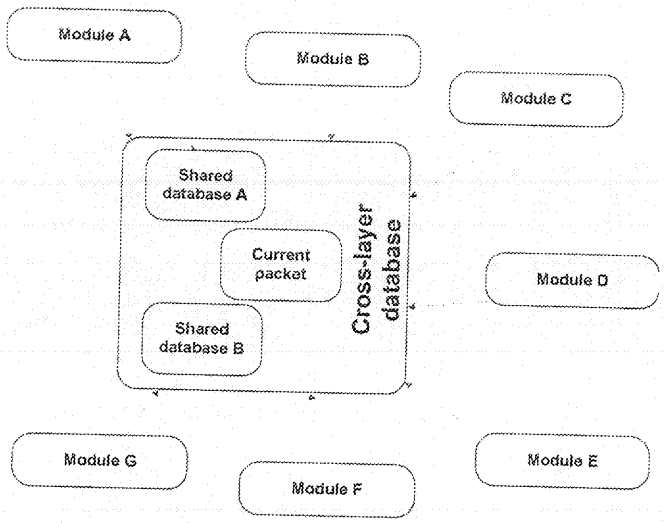


Figure 3. In our modular approach, functionality is divided in modules which interact with a central database

Even though it is possible for the modules to interact directly with each other, this does not promote reusability and interoperability. Therefore, we only allow interaction through a cross-layer database (see figure 3). The shared cross-layer database forms an independent data-repository (represented as a vertical layer in figure 2) which contains several data structures. Modules can register, request or modify the needed parameters through well-defined interfaces. Thus, modules can interact with stored packets, can register to variables that are of interest to them and even create new variables or data structures to share [21] with other modules. Finally, the cross-layer database can have provisions for indicating which parameters are available.

Using the cross-layer data-repository, modules can work together to fulfill advanced network functions. As an example, we give a possible interaction of the QoS module with some functional modules.

- Several routing protocols exist which can offer multiple next-hop candidates (for example Directed Diffusion, [22]). The QoS module can request these from the database and can select the most energy-efficient, the fastest or the most reliable one.
- Once the optimal next hop has been determined, the QoS module can inform the transmission module of the required priority of the packet. Based on this priority, the transmission module knows the correct buffer for queuing the packet.
- Finally, the QoS module can request the MAC module to reduce the transmission delay by selecting a lower back-off value for high-priority packets.

Some examples of parameters which could (and probably should) be made available are the following:

- For each neighboring node, the wake-up times, its position, its mobility information, the associated link quality, its remaining battery power and its role (lightweight, advanced or computing node) should be made available.
- For each QoS class, the required reliability, the maximum delay and the expected traffic load should be known.
- For each current packet, its QoS class, the next-hop candidates, the number of hops it has traversed and the time it has traveled.

A more detailed study of cross-layer exchanges in sensor networks will follow in later work.

B. Modular Heterogeneous Sensornetwork Architecture (Mohesa)

Mohesa forms the core of the universal modular framework. It contains the modules which implement the network protocols. We discern two major types of modules: functional and regulating modules. Both types are depicted in the framework in figure 2.

Functional modules are modules which are called when there are incoming or outgoing packets in the system. They are needed to either actively process the data flow (such as a MAC module, a data aggregation module or an error check module) or to set up a correct processing state for the packets (such as setting the transmission power or checking QoS restraints). Thus, functional modules are called by a scheduler whenever a packets arrives at a node or whenever the node has packets to transmit.

Regulating modules implement protocols which do not actively process the data flow, but instead are needed for a correct internal state of the node. They are called regularly, independently of any actual packet processing. Typical examples are synchronization, position discovery or energy management modules.

The difference between both types of modules is schematically shown in figure 4.

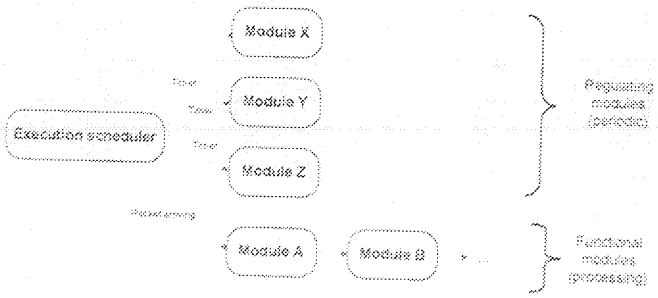


Figure 4. Functional modules are called upon receiving or sending a packet, regulating modules are called regularly

1) Functional modules: We illustrate the principal elements with some example modules, starting with some functional ones.

- The MAC module in lightweight nodes can be very simple: packets are sent using a simple CSMA/CA mechanism. The MAC module in advanced nodes is more advanced. For example, it can be responsible for generating time slots in which lightweight nodes can send their packets. Other possible features include giving priority to some packets, changing its wake-up scheme according to the traffic rate and required delay, or implementing some form of fairness.
- Each node will have modules taking care of transmission control. The basic modules will handle segmentation and error correction. More advanced nodes can also be equipped with an advanced module which can buffer packets for batch transmission. Finally, computing nodes can take care of complex transmission control functions, such as in-order delivery and congestion control.
- The routing module in lightweight nodes can be very simple, e.g. just forwarding all generated data packets to a nearby advanced or computing node. Advanced nodes implement full-routing capabilities: they can create routes based on metrics such as hop distance or remaining battery power. Finally, computing nodes can have functionalities for translating and routing the sensor network packets to an IP-backbone.
- Some modules are only implemented in sufficient capable nodes. For example, the computing nodes may provide QoS by examining incoming packets, adjusting their priority based on the time they are on their way and the distance they still have to travel. To fulfill reliability constraints, packets can be duplicated to a second route. Intermediate, non-computing nodes only have to look at the packet priority, regardless of the required end-to-end QoS. Thus, services such as QoS can be offered to the application even if only a small subset of the nodes are computing nodes.
- Several routing protocols exist where multiple next-hop candidates are stored in the routing table. After these candidates are selected by the routing module, a load balancing module can be used to further optimize the routing of the packets. If the node is capable, the load balancing module can be replaced by a fully functional QoS module.

The order in which modules are called is handled by an execution scheduler. The execution scheduler contains several call sequences. Depending on the situation, the appropriate call sequence is initiated. More advanced nodes will contain more advanced call sequences to handle more complex tasks. In figure 5 an example call sequence is given for handling locally generated packets in the different types of nodes.

2) Regulating modules: The second type of modules, which do not actually process packets, but instead influence the behavior of other modules, are called regulating modules. The regulating functions implemented by the different types of nodes are shown in table I. Advanced regulating functions are typically found in computing nodes: they limit the amount of time the node can spend in a sleeping state.

Type of node	Implemented functionality
Lightweight node	Neighbor Discovery Basic Power Management Basic Energy Management
Advanced node	Neighbor Discovery Synchronization Advanced Power Management Advanced Energy Management
Computing node	Neighbor Discovery Synchronization Advanced Power Management Advanced Energy Management Network Monitoring Position Discovery

Table I
SOME REGULATING MODULES IN THE DIFFERENT TYPES OF NODES

There are many other possible regulating modules. For example, a topology control module could be developed. This module requests the discovered neighbors and, based on the estimated signal-to-noise ratio, selects a subset from those to which no connection may be made. The undesirable nodes can then be hidden by updating the cross-layer database with the new neighbor information.

A last example of a regulating module is a role assignment module. The exact role of a node (lightweight, advanced or computing node) can be determined at design time, but can also be determined based on the neighboring nodes. For example, when an overabundance of advanced nodes is detected, some advanced nodes can switch to a lightweight modus, provided that the necessary basic modules are available.

Finally, note that some functionality can be implemented in both regulating modules and functional modules. For example: a node can contain a regulating module for managing global QoS and a functional module where QoS can be monitored per packet.

C. Application layer

Mohesa provides a common application interface towards the application layer. This will ease application development as application designers do not need to know the specific characteristics of the underlying sensor network [23]. The uniform interface will guarantee a smooth interaction between applications and network modules. This interaction is necessary because some modules have a strong need for interacting both with other modules and with the application layer. Some examples of these interactions are given below:

- In many scenarios, the method of data aggregation should be dictated by the application layer. Depending on the application, data aggregation can be as simple as calculating the average of several data sets or realize complex data processing. This requires a strong interaction of the data aggregation module with the application layer. On the other hand, also the QoS module should be informed by the data aggregation about the amount of introduced delay.
- Localization is needed by many applications. It can be used for tracking purposes or decisions in the application layer, or for routing purposes. The position of the node

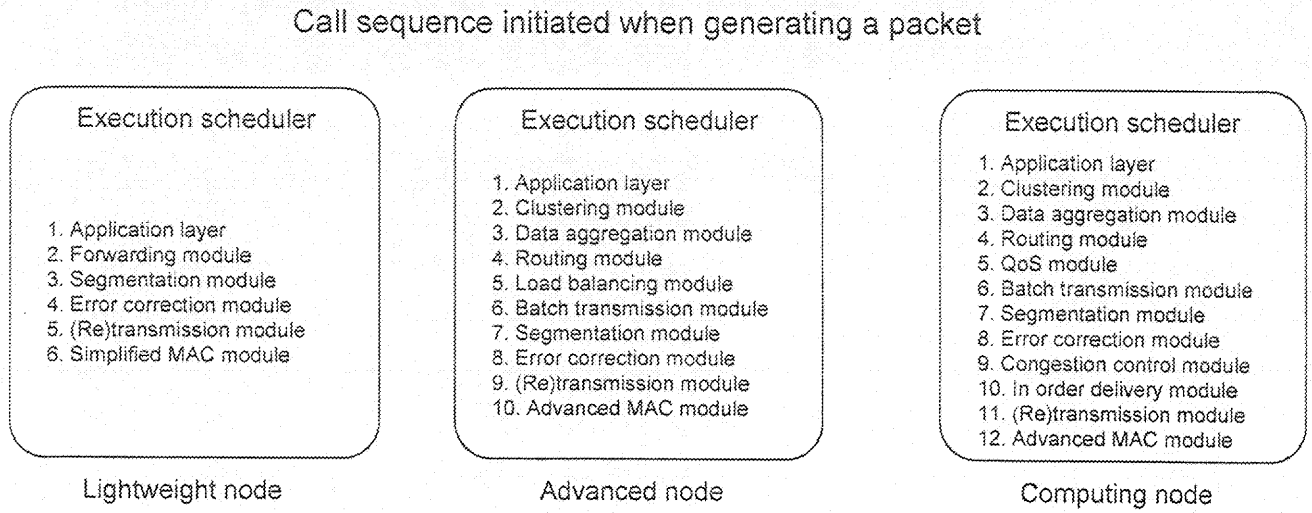


Figure 5. Example call sequence of functional nodes in the different node types

may be calculated using GPS, controlled by the application, or based on radio information. As such, there is a strong need for interaction of the localization module with the application layer and the network modules.

D. Physical layer

The physical layer is a separate layer as its properties largely depend on the design of the hardware. In order to cope with different hardware nodes, the interface between the modular framework and the physical layer should be clearly defined. This is done at the 'Physical Interface'. This Physical Interface can be extended to support various types of physical layers but still provides a common interface to Mohesa. The importance of interaction of other layers with the physical layer is profound as it results in strong optimizations of the protocols.

VII. Coping with interdependencies between modules

In a truly universal framework, any kind of protocol can be integrated. However, this may lead to problems regarding interoperability. For example, a routing protocol may be developed which assumes that all nodes are synchronized or a QoS module may depend on the possibility of the MAC protocol to overhear packets for other destinations. Thus, there is clearly a need to verify and control dependencies between modules.

We envisage two main approaches to ensure compatibility between modules. The first approach uses profiles consisting of compatible modules. The second approach is based on the characteristics of modules: one can either use a descriptive method where each of the individual characteristics are described, or one can classify protocols into groups with similar characteristics.

The first method is through the use of standardized profiles. These contain sets of modules which are known to be compatible with each other and which specify all known interdependencies. Profiles can either be created by enterprising individuals, or by an official standardization institute. By standardizing profiles for specific application domains, the development of sensor applications can be simplified.

The second method for determining dependencies between modules is based on a descriptive approach. The first way to describe characteristics is by describing a number of possible implementation issues for each type of module.

- Some characteristics of a MAC module are: whether it makes use of duty-cycling (uses sleeping periods), whether it uses contention or is contention-free, whether it is synchronous or asynchronous, whether it makes use of polling, whether is can overhear packets for other destinations, whether it has need of a second communication channel, whether it implements retransmissions, whether it supports broadcasts, ...
- Routing protocols characteristics can be based on the routing methods: data-centric, location based or hierarchical. Other characteristics can indicate whether the routing module supports mobility or path recovery, in which way QoS is supported, etc.

When every module indicates the characteristics that are needed for there well-functioning, interdependencies between modules can be verified and controlled.

The second way to use the descriptive approach is by using classification methods. Several efforts have been made to classify similar protocols into a hierarchical tree structure [24], [25]. Similar protocols can be categorized in the same leaf of the tree, making it obvious which protocols have the same characteristics. A syntax can be developed whereby each module has a generic classification number indicating the function of the module, and more specific number indicating the characteristics of the protocol (see figure 6). For example: routing protocol B needs the presence of a synchronized MAC module (1.1.1.*) or a MAC module designed to specifically work with this routing protocol (1.1.4.5). Furthermore, since this routing protocol already includes data-aggregation, no

other data aggregation module (6.*) may be implemented. The requirements for routing protocol B then become: (1.2.* OR 1.3.5.1) AND (NOT 6.*). One disadvantage of the classification method using trees is that it is difficult to categorize new hybrid protocols which do not neatly fit into one of the developed classes. As such, the descriptive approach based on characteristics may be better suited for an innovative domain such as wireless sensor networks.

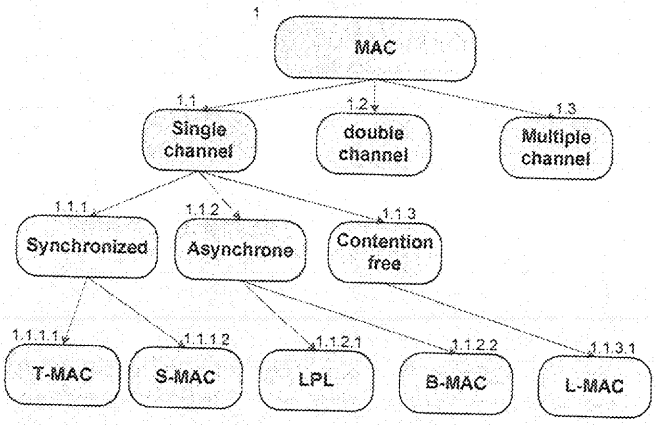


Figure 6. An example classification of MAC modules with there corresponding classification number

VIII. Future work

We have explored several design issues and implementation possibilities for building a modular sensor network architecture. The framework presented here is only a preliminary proposal towards a universal sensor network architecture. Several challenges must still be overcome before we can truly speak of an overall sensor network architecture.

In order to promote the development of new protocols, protocol designers need to analyze the other parts of the system at a fairly high level. Thus, the cross-layer framework should process several node, packet and link statistics and present them at the right level of abstraction. Future work will focus on possible methods for this information exchange. The information needs of several network protocols will be thoroughly analyzed and based on this information, the cross-layer interface will be defined and finally put to test with several classes of protocols.

We will further investigate whether there are any performance penalties in going with a universal framework rather than an implementation tailored to the application. We will also investigate the amount of overhead introduced by the use of the UMF. We presume that the additional overhead of the cross-layer interface will be compensated by the reduction of duplicate functionalities and storage of data structures. This would be in line with similar research, which concludes that the incurred overhead is minimal [16]. More research will be needed before any final conclusions can be made.

Future research will also focus on designing modular network protocols which are optimized for heterogeneous networks. In particular, we are interested in finding ways to exploit the additional capabilities of computing nodes in network consisting mainly of simple lightweight nodes.

Our research will also comprise ways to autonomously assign nodes to classes of comparable capabilities and characteristics. Furthermore, the framework could be expanded with a role-assignment module, which decides at runtime which modules are activated, or which version of the module should be used. Lastly, the feasibility of the framework will be experimentally validated in a heterogeneous hardware testbed.

Finally, our framework does not address any security related aspects. Security could be integrated using an independent method at MAC, routing and application level. However, we envisage the addition of a security module, which could use the cross-layer framework to provide a systemwide level of security.

IX. Conclusion

A lot of additional research is needed before sensor networks can be widely adopted. Current work mostly focuses on scalable and energy-efficient protocols. Although this research is very valuable for the development of sensor networks, there is at present no optimized architecture in which these solutions can be implemented. Support for energy-efficiency and heterogeneity has to be implemented at an architectural level, not only in isolated protocols that do not take the layer interaction into account. Furthermore, in order to promote interoperability of protocols and to reduce the cost of application development for sensor networks, there is a strong need for an application independent solution.

Therefore, we propose a universal modular framework useful for a wide range of applications. The modular design has several key advantages:

- Duplication of functionality is prevented;
- Due to the possibility of cross-module information exchange more energy-efficient protocols are supported;
- Heterogeneity is promoted: modules can be added to a node according to its capabilities;
- Through the replacement of modules, it is easy to adapt to changing network conditions and future developments.

We have explored several ways for protocols to interact with each other. We are convinced that the use of a cross-layer interface can provide sufficient abstraction of the inner working of protocols to allow for effective cooperation between different modules.

Also, we have shown that several requirements for future sensor networks, such as QoS, mobility and energy-efficiency, cannot be supported at a single protocol level. These functionalities should be implemented in dedicated protocols, which require a strong interaction with other protocols, such as routing and MAC protocols. These interactions are fully supported by the proposed modular framework.

Finally, we are convinced that networks will become more and more heterogeneous, and that a universal sensor network architecture should support this evolution.

The use of a modular framework is thus a very promising approach for sensor networks.

REFERENCES

[1] Akyildiz I., Su W., Sankarasubramaniam Y. and Cayirci S., "A survey on sensor networks." IEEE Communications Magazine, Vol. 40, No. 8, August 2002, pp. 102–114

[2] Akyildiz I., Kasimoglu, I.H., "Wireless Sensor and Actor Networks: Research Challenges." Ad Hoc Network Journal (Elsevier), Vol. 2, No. 4, pp. 351–367, October 2004

[3] Stojmenovic, I., "Handbook of Sensor Networks: algorithms and architectures", Wiley-Interscience, ISBN: 0471684724, December 2005

[4] Srivastava, V.; Motani, M., "Cross-layer design: a survey and the road ahead," Communications Magazine, IEEE , vol.43, no.12, Dec. 2005, pp. 112–119

[5] Chlamtac, I., Conti, M. and Liu, J., "Mobile Ad Hoc Networks: Imperatives and Challenges", Elsevier Ad Hoc Networks, Vol. 1, No. 1, July 2003, pp. 13-64

[6] Y. Zhang, L. Cheng, "Cross-layer Optimization for sensor networks." New York Metro Arena Workshop 2003, New York, NY, September 2003

[7] Toumpis, S. , Goldsmith, A. J., "Performance, optimization, and cross-layer design of media access protocols for wireless ad hoc networks," ICC 2003 - IEEE International Conference on Communications, May 2003 pp. 2234-2240

[8] Conti, M., Maselli, G. and Turi, G., "Cross-Layering in Mobile Ad Hoc Network Design", IEEE Computer, Vol. 37, No. 2, February 2004, pp. 48-51

[9] Shakkottai, S., Rappaport, T.S., Karlsson, P.C., "Cross-layer design for Wireless Networks," IEEE Communication Magazine, vol. 41, no. 10, Oct. 2003, pp. 74–80

[10] Melodia, T., Vuran, M. C., Pompili, D., "The State of the Art in Cross-layer Design for Wireless Sensor Networks," in Proceedings of EuroNGI Workshops on Wireless and Mobility, Springer Lecture Notes on Computer Science, LNCS 388, Como, Italy, July 2005

[11] Marrón, P. J., Lachenmann, A., Minder, D., Hahner, J., Rothermel, K. and Becker, C., "Adaptation and cross-layer issues in sensor networks," in Proc. of the First International Conference on Intelligent Sensors, Sensor Networks & Information Processing (ISSNIP 2004), pp. 623-628, December 2004.

[12] van Hoesel, L. Nieberg, T. J. W. Havinga, and P.J.M., "Prolonging the lifetime of wireless sensor networks by cross-layer interaction," Wireless Communications, IEEE, vol. 11, Issue 6, pp. 78- 86, Dec. 2004.

[13] H. Kwon, T. H. Kim, S. Choi, and B. G. Lee, "A Cross-Layer Strategy for Energy-Efficient Reliable Delivery in Wireless Sensor Networks," Wireless Communications, IEEE Transactions on, Vol. 5, Iss. 12, December 2006, pp. 3689-3699

[14] Braden, R., Faber, T., and Handley, M. "From protocol stack to protocol heap: role-based architecture," SIGCOMM Comput. Commun. Rev. 33, no. 1, Jan. 2003, pp. 17–22

[15] Culler, D., Dutta, P., Ee, C.T., Fonseca, R., Hui, J., Levis, P., Polastre, J., Shenker, S., Stoica, I., Tolle, G. and Zhao, J., "Towards a Sensor Network Architecture: Lowering the Waistline." In Proceedings of the Tenth Workshop on Hot Topics in Operating Systems (HotOS X), 2005.

[16] Polastre, J., Hui, J., Levis, P., Zhao, J., Culler, D., Shenker, S., and Stoica, I. "A unifying link abstraction for wireless sensor networks. " SenSys '05, San Diego, CA, USA, Nov. 2005, pp. 76–89

[17] Ee, C.E., Fonseca, R., Kim, S., Moon, D., Tavakoli, A., Culler, D., Shenker, S. and Stoica, I. "A Modular Network Layer for Sensornets." In the Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation (OSDI 2006), Seattle, WA, November 2006

[18] TinyOS operating system, http://www.tinyos.net/

[19] SOS operating system, https://projects.nesl.ucla.edu/public/sos-2x/

[20] Kawadia, V.; Kumar, P.R., "A cautionary perspective on cross-layer design," Wireless Communications, IEEE , vol.12, no.1, Feb. 2005, pp. 3- 11

[21] Lachenmann, A., Marrón, P. J., Minder, D., Gauger, M., Saukh, O. and Rothermel, K., "TinyXXL: Language and Runtime Support for Cross-Layer Interactions," Proc. of the Third Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, Reston, VA, USA, September 2006, pp. 178–187

[22] Intanagonwiwat, C., Govindan, R. and Estrin, D., "Directed diffusion: a scalable and robust communication paradigm for sensor networks. " Mobile Computing and Networking, 2000, pp. 56–67

[23] Romer, K., Kasten, O., Mattern, F., "Middleware Challenges for Wireless Sensor Networks," Mobile Computing and Communications Review, Vol. 6, Number 2, October 2002

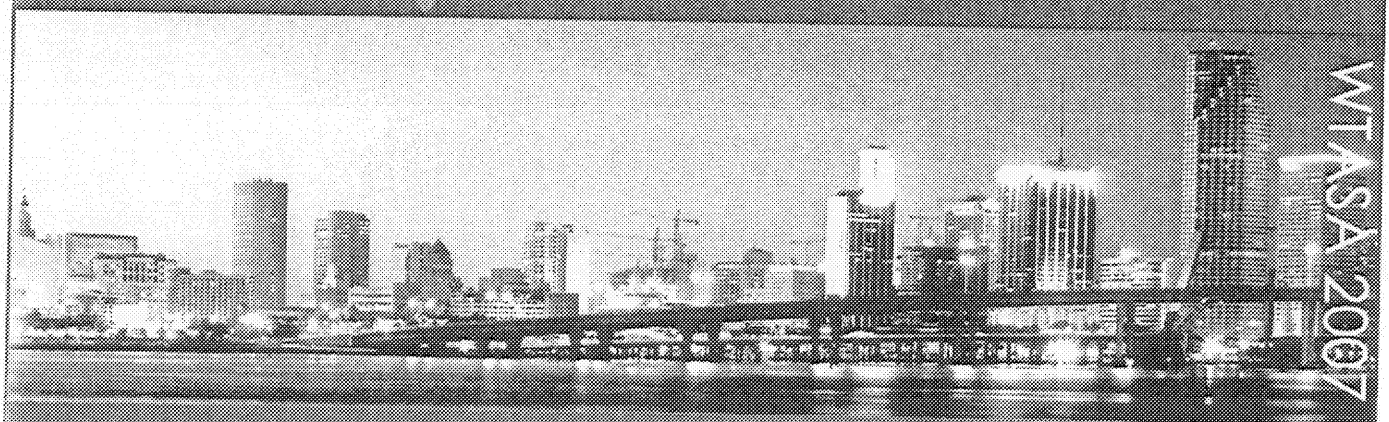
[24] The Mac Alphabet Soup: a MAC protocol taxonomy. https://ipanwo.st.ewi.tudelft.nl/ koen/MACsoup/taxonomy.php

[25] K. Akkaya, K., Younis, M. "A survey of routing protocols in wireless sensor networks," Elsevier Ad Hoc Network Journal, Vol. 3, no. 3, 2005, pp 325–349

Proceedings of International Workshop on Theoretical and Algorithmic Aspects of Sensor and Ad-hoc Networks

Miami, Florida
June 28-29, 2007

Edited by
S. Kami Makki, University of Toledo
Xiang-Yang Li, Illinois Institute of Technology
Niki Pissinou, Florida International University
Shamila Makki, Florida International University
Masoumeh Karimi, Florida International University



WTASA 2007

Sponsored by National Science Foundation (NSF)